

Processes can communicate through a shared resource

- An area of memory
- A file

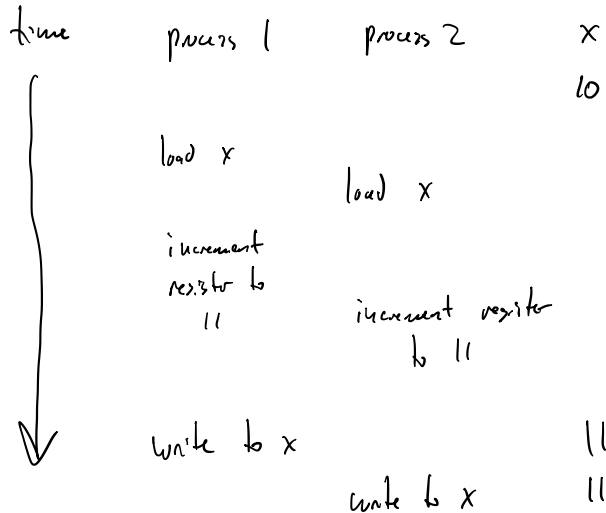
Creating a new process

- `fork()` system call in POSIX
 - Child process is created, which is a copy of the parent process
 - All segments are copied: stack, heap, etc.
 - All registers are copied
 - Child begins executing directly after the `fork()` call
 - `fork()` returns 0 for the child, and the PID of the child for the parent

Race condition

- Two or more processes are reading and/or writing shared data and the result depends on the order of events

$$x = x + 1$$



Mutual exclusion

- If one process is using a shared resource, other processes will be excluded from doing the same thing

Critical region

- Part of the program where a shared resource is accessed

Atomic action

- An action that is carried out in its entirety without interruption even if it involves multiple instructions

Semaphore

- A variable that stores an integer
- Two operations: down and up (wait and post in POSIX)
- Down waits until the semaphore is non-zero, then decrements it in an atomic action
- Up increments the semaphore